HW #3: Motion Planning with RRT

Name: YOUR NAME HERE

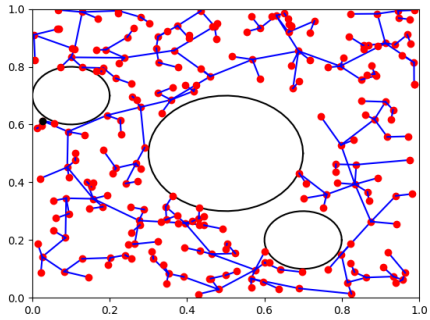
Deliverable: PDF write-up and code by **Friday October 29th, 5:00pm**. Your PDF should be generated by replacing the placeholder images in this LaTeX document with the appropriate solution images for each question. Your PDF and code (hw3_rrt.py) is to be submitted via the course Canvas. The scripts will automatically generate the appropriate images, so you only need to recompile the LaTeX document to populate it with content.

Graduate Students: You are expected to complete the entire assignment.
Undergraduate Students: You need only complete questions that do not have **(GRAD)** next to them.

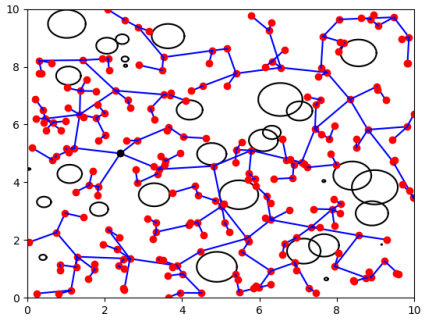
To generate figures, you will need to have matplotlib:
pip install matplotlib (you should have this from your previous assignment)

1. Rapidly-Exploring Random Trees

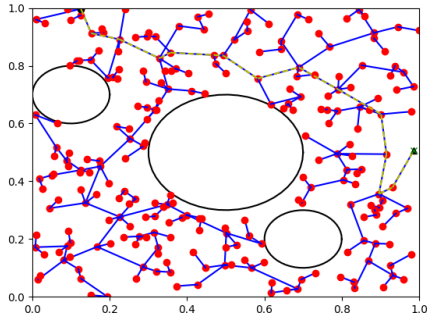
(a) **[30pts] RRT.** First, you will implement the RRT algorithm for holonomic robots on the X-Y plane. You will need to put your code in `code/hw3_rrt.py` in the function `rrt`. You will also need to add your own code to implement the general-purpose `get_nearest_vertex` and `steer_holonomic` functions (any valid implementation will do). Make sure the figures generated from the two provided 2D obstacle worlds look reasonable and appear below.



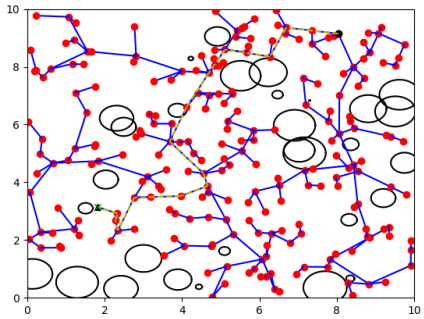
(a) RRT from the First 2D Domain



(b) RRT on a Random 2D Domain



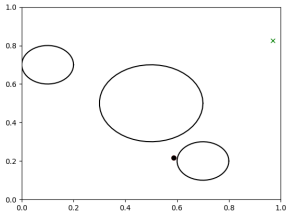
(a) Goal-Directed RRT on the First 2D Domain



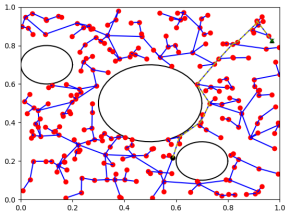
(b) Goal-Directed RRT on a Random 2D Domain

(b) **[50pts] RRT*.** Implement the RRT* algorithm for holonomic robots on the X-Y plane. You will need to put your code in `code/hw3_rrt.py` in the function `rrt_star`. Make sure the figures generated from the two provided 2D obstacle worlds look reasonable and appear below.

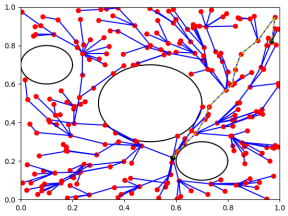
(c) **[20pts] Discrete Non-Holonomic RRT (GRAD).** If our robot cannot freely traverse the coordinate system used in our RRT algorithm, then we can't just add vertices wherever we want and interpolate from existing ones (our robot may not be able to take that path or get there!). Fill in the code for the *steer_discrete_non_holonomic* function that would operate on a discrete action space given as a list *actions_list*. This agent will only be able to move diagonally!



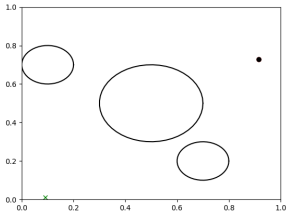
(a) Planning Domain



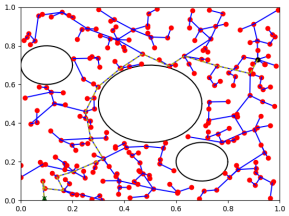
(b) RRT on the First 2D Domain



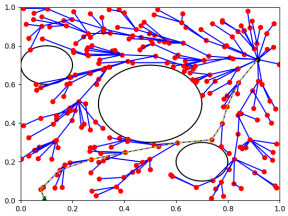
(c) RRT* on the First 2D Domain



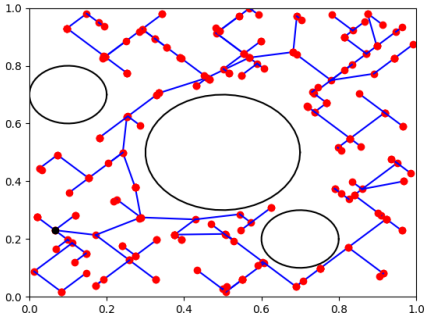
(a) Planning Domain



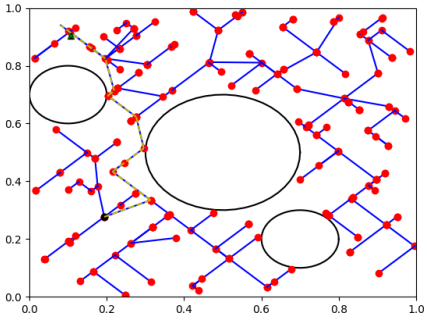
(b) RRT on the Second 2D Domain



(c) RRT* on the Second 2D Domain



(a) RRT with discrete action space



(b) RRT with discrete action space